

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-028166

(43)Date of publication of application : 04.02.1994

(51)Int.Cl.

G06F 9/06

(21)Application number : 04-178516

(71)Applicant : TOSHIBA CORP

(22)Date of filing : 06.07.1992

(72)Inventor : FUJITA TORU

(54) TEXT INPUT/OUTPUT SYSTEM

(57)Abstract:

PURPOSE: To enable a designer to speedily and easily input a program source in his coding style, to easily share design data, and to improve designing efficiency by automatically converting names in the program according to a naming rule.

CONSTITUTION: A variable 'A' (figure A) as an integer type linear array are converted into a name 'ALst' (figure B) according to the naming rule. A character type variable 'Knam' and a character type pointer 'Snam' are converted into 'KnamLtr' and 'SnamLtrPtr' respectively. A record type variable 'Data' is converted into 'DataStr' and the members constituting the record type are also converted according to the naming rule.

(A)

```

<int> A[5];
<char> Data; <struct>
<record> Data{
    <int> X; Y;
    <char> Idet[5,5];
};

```

(B)

```

<int> ALst[5];
<struct> KnamLtr; SnamLtrPtr;
<record> DataStr{
    <int> X; Y;
    <char> Idet[5,5];
};

```

LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁(JP)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平6-28166

(43)公開日 平成6年(1994)2月4日

(51)Int.Cl.⁵

G 0 6 F 9/06

識別記号

庁内整理番号

F I

技術表示箇所

4 3 0 M 9367-5B

審査請求 未請求 請求項の数1(全13頁)

(21)出願番号 特願平4-178516

(22)出願日 平成4年(1992)7月6日

(71)出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(72)発明者 藤田 透

神奈川県川崎市幸区小向東芝町1 株式会
社東芝総合研究所内

(74)代理人 弁理士 三好 秀和 (外1名)

(54)【発明の名称】 テキスト入出力システム

(57)【要約】

【目的】 命名規則に従って自動的にプログラム上の名前の変換を行う。

【構成】 ソースコード1上の、整数型の1次元配列である変数“A”(図(A))は、命名規則に従って“A L s t”という名前に変換される(図(B))。文字型の変数である“K n a m”、文字型のポインタである“S n a m”はそれぞれ“K n a m L t r”、“S n a m L t r P t r”という名前に変換される。レコード型の変数である“D a t a”は“D a t a S t r”に変換され、そのレコード型に含まれるメンバーも命名規則に従って名前が変換される。

【効果】 設計者が自らのコーディングスタイルで迅速かつ容易にプログラムソースの入力を行える。設計データの共有化が容易で、設計効率が良くなる。

(A)

```
<int> A[5];  
<char> Knam, *Snam;  
<record> Data{  
    <int> X, Y;  
    <char> Ident[5.3];  
};
```

(B)

```
<int> ALst[5];  
<char> KnamLtr, SnamLtrPtr;  
<record> DataStr{  
    <int> X, Y;  
    <char> IdentLtrTab[5.6];  
};
```

【特許請求の範囲】

【請求項 1】 プログラム上で用いられる名前を新たな名前に変換する際の規則を保存している名前変換規則保存手段と、

この名前変換規則保存手段に保存されている名前変換規則に従ってプログラム上の名前を新たな名前に変換する名前変換手段と、

変換される前後の名前を対応付けて格納する名前格納手段とを備えたことを特徴とするテキスト入出力システム。

【発明の詳細な説明】**【0001】**

【産業上の利用分野】 本発明は、プログラム上の名前の変換を自動的に行うことができるテキスト入出力システムに関する。

【0002】

【従来の技術】 プログラム開発などのように複数の設計者が共同作業を行うプロジェクトでは開発者相互のコミュニケーションやソースプログラムのやりとりをスムーズにする必要がある。そのために、プログラム上で用いられるユーザ名、ファイル名、あるいは変数名などの命名規則を設定し、複数の設計者に共通した名前に変換してコーディングを行っていた。

【0003】

【発明が解決しようとする課題】 しかしながら、名前を変換する際、設計者は常に規則を念頭においてコーディングを行わなくてはならないという煩わしさがあつた。また、自らのコーディングスタイルと他の設計者のコーディングスタイルとの違いから、作業の効率低下を引き起こすという問題があつた。

【0004】 本発明は上述した従来の共同プロジェクトでのコーディング・設計データ管理の欠点を解決するために、プロジェクト管理のために定めた命名規則に従って自動的にプログラム上の名前の変換を行うことによって設計者が従来通りの自らのコーディングスタイルで迅速かつ容易にプログラムソースの入力を行うことができるテキスト入出力システムを提供するものである。

【0005】

【課題を解決するための手段】 上記目的を達成するために、本発明のテキスト入出力システムは、プログラム上で用いられる名前を新たな名前に変換する際の規則を保存している名前変換規則保存手段と、この名前変換規則保存手段に保存されている名前変換規則に従ってプログラム上の名前を新たな名前に変換する名前変換手段と、変換される前後の名前を対応付けて格納する名前格納手段とから構成されている。

【0006】

【作用】 上記構成により、本発明は、名前変換手段が名前変換規則保存手段に保存されている規則を読み込み、プログラム上で名前を見つけると名前変換規則に従って

その名前を変換して名前格納手段に登録する。登録された名前はソースに埋め込まれ、あるいはソースの表示に利用される。

【0007】

【実施例】 以下、この発明の実施例を図面を参照に説明する。

【0008】 まず、この発明をコンパイラに適用した例について図 1~4 を用いて説明する。

【0009】 図 1 は、本発明を適用したコンパイラの構成例である。

【0010】 同図において、ソースコード 1、字句解析部 2、構文解析部 3、コード生成部 4、保存部 5、及び目的コード 6 は、従来からのコンパイラによる構成である。本発明を適用したコンパイラでは、これらの構成に命名規則指定ファイル 7、命名規則指定管理部 8、制御部 9、名前変換テーブル 10 を追加している。

【0011】 命名規則指定ファイル 7 は、変換規則を保存しているファイルであり、詳細は後述する。

【0012】 命名規則指定管理部 8 は、命名規則指定ファイル 7 に保存されている規則に従って名前を変換するところである。

【0013】 制御部 9 は、命名規則指定管理部 8 などを制御するものである。

【0014】 名前変換テーブル 10 は、変換された前後の名前を対応付けて格納するところである。

【0015】 図 2 は、設計者によって作成された命名規則指定ファイル 7 の具体例である。この例では使用するプログラム言語に整数型、文字型、及び複数のデータ型のセットであるレコード型の 3 つのデータ型を仮定して、それぞれのデータ型に配列やポインタの形式を指定することができるとする。

【0016】 この命名規則指定ファイル 7 は、これらのデータ型、データ形式に右欄の文字列を付加させる指定を行うものである。この例では複数型の変数は変更なし、文字型・レコード型の変数はそれぞれ“L t r”、“S t r”という文字列をその名前の後につけ加える。さらに変数のデータ形式によって 1 次元・2 次元の配列にはそれぞれ“L s t”、“T a b”、ポインタには“P t r”を名前の後につけ加える。

【0017】 このように、この発明を適用したコンパイラは構成されており、次にこのコンパイラの動作を説明する。

【0018】 字句解析部 2 がプログラムのソースコード 1 を字句単位で読み込む。読み込まれた字句のうちでプログラム言語の仕様により名前（例えば、定数名、関数名、変数名など）であると判断されたものは命名規則指定管理部 8 に渡されて命名規則に従って変換され、命名規則指定管理部 8 から字句解析部 2 に返される。

【0019】 図 3 は、読み込まれ変換される前後のソースコードの例である。図（A）が変換前のソースコー

ド、(B)が変換後のソースコードである。この例では整数型の1次元配列である変数“A”は命名規則に従って“ALst”という名前に変換されている。以下同様にして文字型の変数である“Knam”、文字型のポインタである“Snam”はそれぞれ“KnamLtr”、“SnamLtrPtr”という名前に変換される。レコード型の変数である“Data”は“DataStr”に変換され、そのレコード型に含まれるメンバーも命名規則に従って名前が変換される。

【0020】また命名規則指定管理部8は、変換前後の名前を名前変換テーブル10に登録する。図4は名前変換テーブルの例である。この例では変換前後の変数名・変数のデータ型とデータ形式を保存している。このテーブルの作成後は、ソースコードに現れる名前の変換はこのテーブルを参照して行なわれる。

【0021】さらに、名前が変換されたソースコード1が構文解析部3・コード生成部4・保存部5と順番に渡されて目的コード6がファイルとして保存される。また制御部9によって名前変換後のソースコード1がファイルに保存される。

【0022】2番目の例として本発明が設計データ管理に適用された例を示す。

【0023】図5は複数の設計者が使用するデータ管理システムの構成例である。

【0024】図5においては、ソースコード1、データ読み込み部11、データ表示部12、データ保存部13、保存ファイル14が従来からの構成であり、ファイル名設定ファイル15、命名規則指定管理部8、制御部9、名前変換テーブル10がこの発明によって追加された機能である。

【0025】命名規則指定管理部8、制御部9、名前変換テーブル10は、図1で示したものと同様の機能を有している。ファイル名設定ファイル15は、図1の命名規則指定ファイル7に相当するものであり、ファイル名の変換規則を保存しているファイルである。

【0026】このように構成されたデータ管理システムでは、データ読み込み部11が、ある名前のファイルに保存されているソースコード1を読み込むと同時に、制御部9がユーザの名前を確認してファイル名設定ファイル15の指定する規則にしたがってファイル名を変換する。

【0027】図6はファイル名設定ファイルの例である。図6のはじめの文字列“PROA”はプロジェクト名である。次の数字はそのプロジェクトに関わるグループの数である。その後は各グループの担当するサブプロジェクト名、グループメンバーの数とグループメンバーの名前である。

【0028】いまサブプロジェクト“CPU”に携わっている“fujita”というメンバーが“ABUS”というファイルを新たに作成すると、データ管理システ

ムはこのファイルのソースコード1を読み込み、ファイル名設定ファイル15を参照してファイル名を“PROA-CPU-ABUS-fujita”に変換する。そして、変換前後のファイル名を名前変換テーブル10に登録する。

【0029】これにより、他のメンバーがこのファイルをデータ管理システムから利用しようとする場合には、この変更後の名前でも参照することができる。ファイルの作成者である“fujita”がこのファイルを利用しようとする場合には、制御部9がメンバー名をチェックしてファイル名設定ファイル15を参照し、変更前後のどちらの名前でも参照できる。

【0030】また、データ読み込み部11が名前を読み込んだ時にそれを命名規則指定管理部8に渡し、名前変換テーブル10を利用して変換前の名前に戻して表示することもできる。

【0031】ソースコード1に修正が加わった場合には、制御部9が修正されたことを確認し、データ読み込み部11から必要なデータを受けとって名前変更テーブル10の修正を行なう。このため設計者は、自分のコーディングスタイルで作成したソースコード1のままであると仮定してデータを利用したり、変更することができる。

【0032】3番目の例として本発明がハードウェア設計フレームワークに適用された例を示す。

【0033】図7は、ハードウェア設計フレームワークに本発明を適用したシステムの構成例である。

【0034】図7で示したシステムにおいては、フレームワークシステム16、データ変換部17が従来から備えられている構成であり、命名規則指定管理部8、制御部9、名前変換テーブル10、及びモジュール階層データ18が新たに追加構成されたものである。

【0035】図5の説明の際にも述べたように、命名規則指定管理部8、制御部9、名前変換テーブル10は、図1で示したものと同様の機能を有している。モジュール階層データ18は、図1の命名規則指定ファイル7や図5のファイル名設定ファイル15に相当するものであり、モジュール名の変換規則を保存しているデータファイルである。

【0036】このように構成されたハードウェア設計フレームワークシステムの作用を以下で説明する。

【0037】図8はフレームワークでの設計画面の例である。図8の“SYSA”というシートは一つのモジュールを表しており、モジュール“SYSA”はその中にモジュール“A”、“B”、“MuItP2”などを含んでいる。

【0038】いまモジュール“SYSA”の設計者がモジュール“B”の内部構造を設計し、新たにシート“B”を作成する。その際にシート“SYSA”に含まれているモジュール“MuItP2”をシート“B”の

内部にコピーして使用している。この時点ではモジュール“MultiP2”を表現する設計データは一つしか存在していない。

【0039】フレームワークシステム16は、これらのデータからモジュール階層データ18を抽出する。制御部9は新しいシートが作成されたことが判るとモジュール階層データ18を命名規則指定管理部8に読み込ませる。そしてモジュール階層データ18に基づいて図9のようにモジュール名を変換して名前変換テーブル10を作成する。

【0040】図10は名前変換テーブル10の例である。テーブル1（図（A））を見ると、モジュール“SYSA”の下位モジュールである“A”、“B”、“C”、“MultiP2”はそれぞれ“SYSA-A”、“SYSA-B”、“SYSA-C”、“SYSA-MultiP2”に変換されている。モジュール“SYSA-B”の下位モジュールである“MultiP2”は“SYSA-B-MultiP2”に変換されている。

【0041】設計者が自らのコーディングスタイルに従って自由にモジュール名（ここでは、“A”、“B”、“C”、“MultiP2”）をつけると、自動的にモジュールの階層構造を反映したモジュール名に変換され、ほかの設計者とのデータの共有が容易になる。

【0042】ここで同じ構造を持つモジュールである“SYSA-MultiP2”と“SYSA-B-MultiP2”に別の名前がついているが、モジュールの設計データは“MultiP2”が一つだけ存在していて、フレームワークシステム16が名前変換テーブル10を参照してそれぞれの名前にモジュール設計データを対応させている。

【0043】もしここでモジュール“SYSA-B-MultiP2”の設計データを変更しようとする、制御部9は命名規則指定管理部8に知らせて名前変換テーブル10から、同じ設計データを持つモジュール名“SYSA-B-MultiP2”、“SYSA-MultiP2”を取り出し、設計者にデータの変更をどのモジュールに対して行うかを指定させる。このとき、同じ設計デ

ータを持つすべてのモジュールを指定すれば、データの変更はただ一つの設計データである“MultiP2”に対して行われる。

【0044】またモジュール“SYSA-B-MultiP2”のみを指定した場合には設計データ“MultiP2”に変更の加えられた結果が新たに設計データ“MultiP2-2”として保存され、名前変換テーブル10は図10（B）のテーブル2のように変更される。このためモジュール設計データ量の増大を防ぐことができる。

【0045】

【発明の効果】以上述べてきたように、本発明によれば、設計者は、自らのコーディングスタイルを変更することなしに設計データの作成を行なうことができるため、煩わしさを軽減できるとともに、設計データの共有化が容易となり、効率の良い設計が可能となる。

【図面の簡単な説明】

【図1】本発明をコンパイラに適用した構成例。

【図2】図1で示した命名規則指定ファイルの具体例。

【図3】図1で示したコンパイラによって変換される前後のソースコードの例。

【図4】図1で示した名前変換テーブルの例。

【図5】本発明をデータ管理システムに適用した構成例。

【図6】図5で示したファイル名設定ファイルの例。

【図7】本発明をフレームワークに適用した構成例。

【図8】図7で示したフレームワークで変換される前のモジュール名。

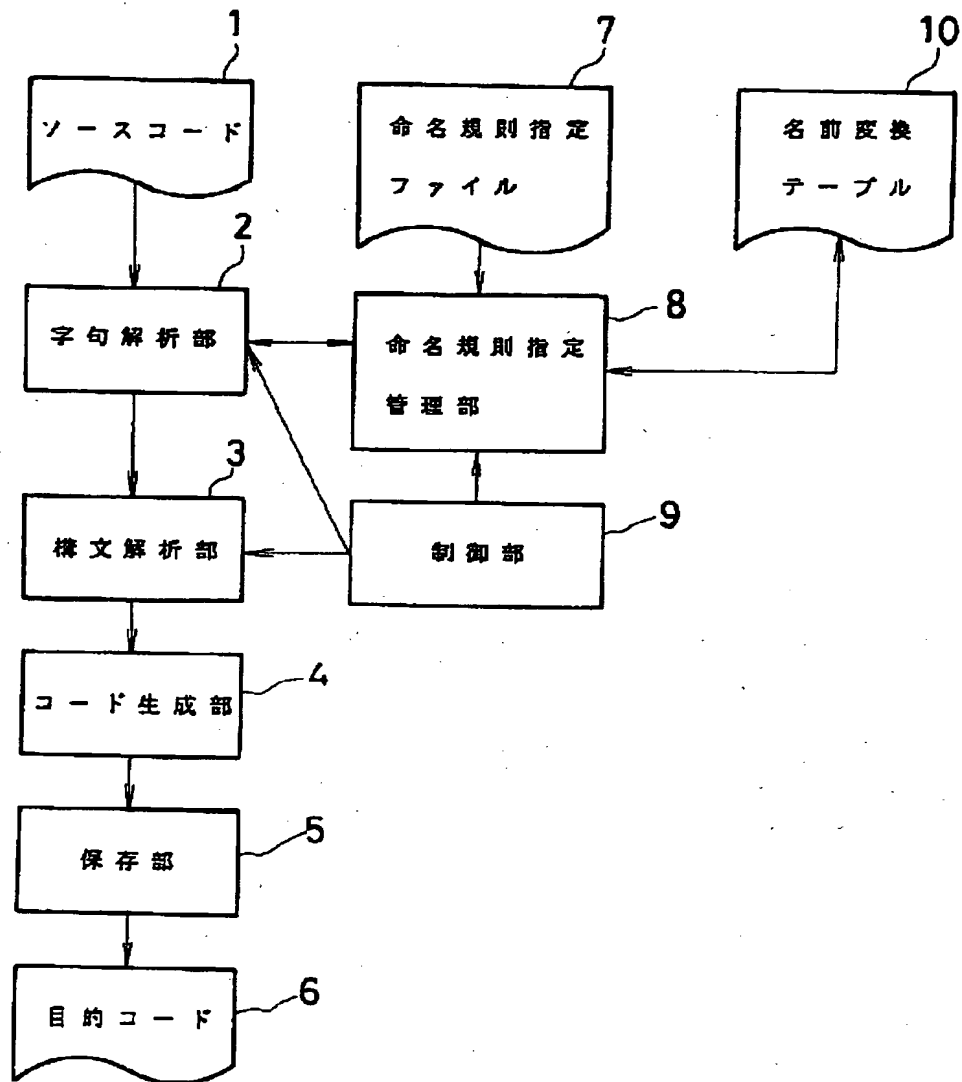
【図9】図7で示したフレームワークで変換された後のモジュール名。

【図10】図7で示した名前変換テーブルの例。

【符号の説明】

- 7 命名規則指定ファイル
- 8 命名規則指定管理部
- 9 制御部
- 10 名前変換テーブル
- 15 ファイル名設定ファイル
- 18 モジュール階層データ

【図1】



【図2】

7

データ型	n	
Integer	" "	1
Character	"Ltr"	2
Record	"Str"	n
<hr/>		
データ形式	n	
Array x 1	"Lst"	1
Array x 2	"Tab"	2
Pointer	"Ptr"	n

【図3】

(A)

```
<int> A[5];  
<char> Knam, *Snam;  
<record> Data{  
    <int> X, Y;  
    <char> Idnt[5.5];  
};
```

(B)

```
<int> ALst[5];  
<char> KnamLtr, SnamLtrPtr;  
<record> DataStr{  
    <int> X, Y;  
    <char> IdntLtrTab[5.5];  
};
```

【図6】

15

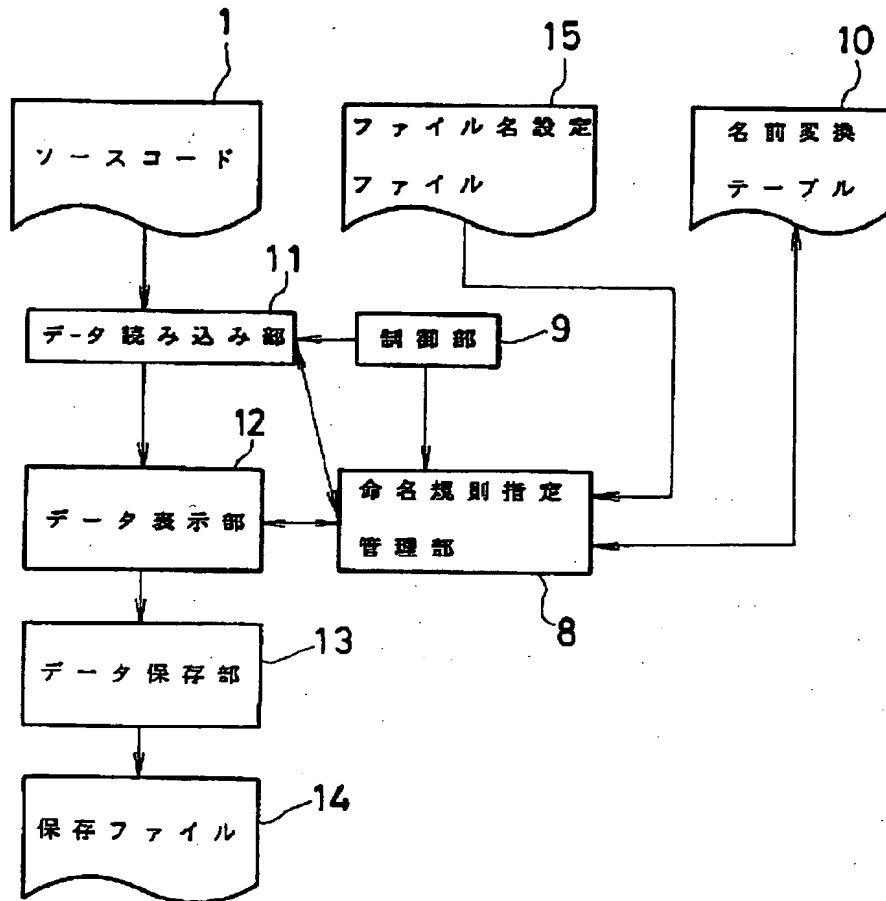
"PROA"
n
"CPU"
2
"fujita"
"sakai"
"BUS"
1
"takahashi"

【図4】

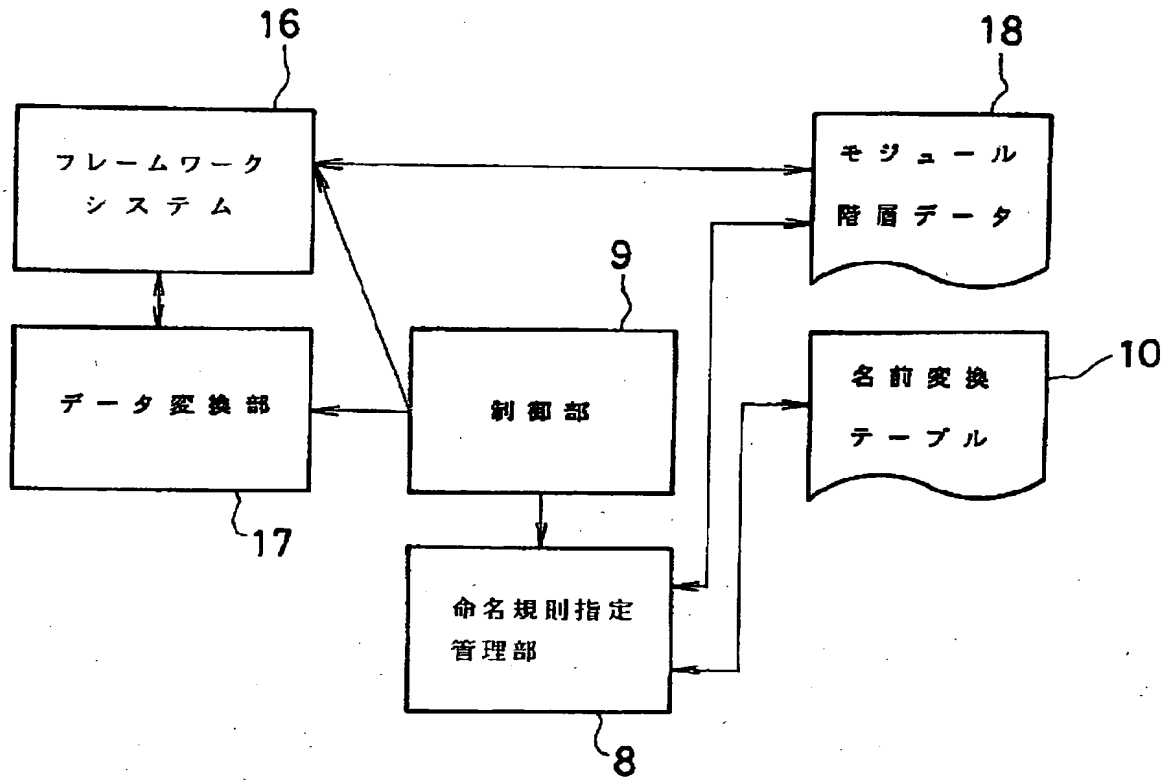
10

"A"	
"ALst"	
int	Array x 1
"Data"	
"DataStr"	
record	
"Idnt"	
"IdntTab"	
char	Array x 2

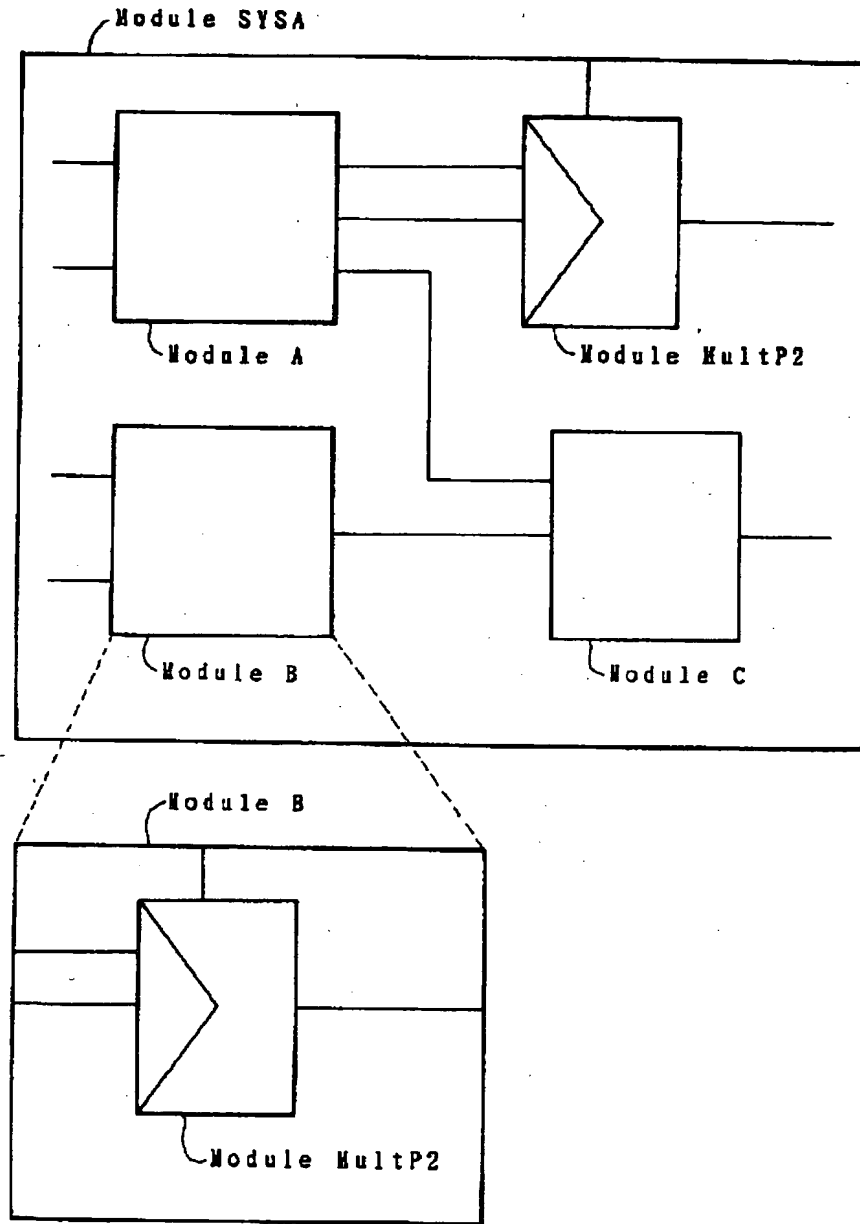
【図5】



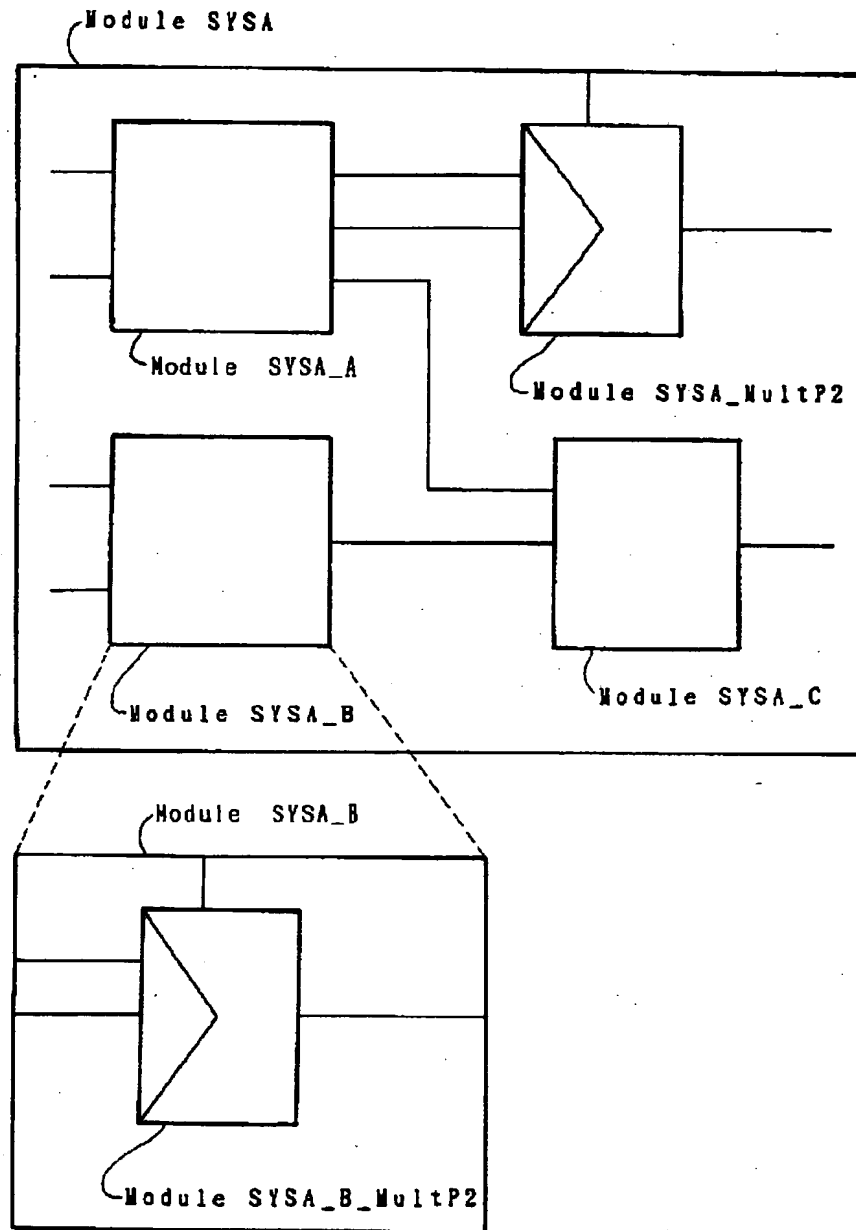
【図7】



【図8】



【図9】



【図10】

(A)

SYSA	1
SYSA	
A	1
SYSA_A	
B	1
SYSA_B	
C	1
SYSA_C	
MultP2	2
SYSA_MultP2	
SYSA_B_MultP2	

テーブル 1

(B)

SYSA	1
SYSA	
A	1
SYSA_A	
B	1
SYSA_B	
C	1
SYSA_C	
MultP2	1
SYSA_MultP2	
MultP2_2	1
SYSA_B_MultP2	

テーブル 2